CSIS 112: Lab 4—Grade Book I

**Instructions**:

Create a class to maintain a *GradeBook*. The class should allow information on up to 3 students to be stored. The information for each student should be encapsulated in a *Student* class and should include the student's last name and up to 5 grades for the student. Note that less than 5 grades may sometimes be stored. Your *GradeBook* class should at least support operations to add a student record to the end of the book (i.e., the book does not have to be in alphabetical order), and to list all student records (name and associated grades) it contains. To add a student to the end of the book, the user should not have to specify a position, (ie. your *GradeBook* should keep track of the last position and add to the end automatically). Your *Student* class should include at least the operations to allow entry of the last name and up to 5 grades, and to return the name and grades for that student.

You should write a main program that creates a grade book and presents a menu to the user that allows them to select either Add (A), or List (L), or Quit (Q). *Add* should allow the user to enter a student record (name and grades) and add it to the end of the Gradebook list. *List* should list all student records currently in the grade book. You should be able to add and list repeatedly, until you select Q to quit.

Use good coding style and principles for all code and input/output formatting. All data in a class must be private. Put each class declaration in its own header file and its implementation in a separate .cpp file.

Helpful hints on how to approach this lab

1. In main():
   1. Prompt the user to enter a name for a gradebook.
   2. Create an object of the gradebook class, passing in the name of the gradebook as an argument to the constructor of the Gradebook class.
   3. Create a menu with “cout” statements.
   4. Create a switch statement that contains the cases that the user could enter.
   5. For the Add students case, prompt the user to enter a name for the student if the number of students already existing is less than 3.

1. Using the Gradebook object, call the function in the Gradebook class to add a student, passing in the name of the student that the user entered.

* 1. For the List students case, call the function that prints student records, again by using the Gradebook object to call the function in the gradebook class.

1. In the Gradebook class:
   1. Make sure you have a constructor that initializes the gradebook name by calling a Set function in the class.  Remember that the name is passed in from main().
   2. For all of the data members you have in your class, be sure to have a Set and Get function for each.
   3. The Gradebook class contains the function to add a Student.  In it, you will create a new student object (passing in the name of the student that came from main()), and add it to the vector of Student objects (which is a data member in the Gradebook class).
   4. As soon as you create a student object in the Gradebook class and add it to the vector, call the function in the Student Class that allows the user to enter up to 5 grades for that student.
   5. Include a function that will print each student object in the Gradebook class.  That is, loop over all the student objects in the vector and for each object, call a function in the Student class that will print each student record.  It’s like using Gradebook as a middleman… main() calls the Gradebook function to list the students.  Gradebook contains the loop to iterate over each student object in its vector, then within the loop, each student object calls a function in the Student class that will print out the names and grades of a given student object.
2. In the Student class:
   1. Make a constructor that initializes the student’s name to whatever was entered in main() and subsequently passed in to your Gradebook function that adds new students.  Recall that this information will be passed in to the Student constructor from the Add Student function in Gradebook.
   2. Create a function that will allow the user to enter up to 5 grades for a student.  This function will actually contain the cout and cin commands to prompt the user to enter each grade (Don’t worry about putting error checks in here because I won’t try to break it.)
   3. Naturally, this class will contain either an array or vector (your choice) of grades that the user will enter (no more than 5).
   4. This class will contain a function that displays the grades for a student.  It will need to loop over all the grades in the array for the student and print them out.  This function will be called from the function in the Gradebook class that prints out each student object.

**Deliverables**:

* Complete the programming assignment described above and submit your completed assignment in accordance with the lab submission policies.